IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: **Lorenzo Di Gregorio al.**

Serial No.: **10/719,794**

Filing Date: **November 21, 2003**

Title: **Device for Controlling Processing of Data Elements of a Data System**

Group Art Unit: **2183**

Examiner:

Attny. Docket No. **068758.0146**
Client Ref.: **IO387US/MGL/pp**

Mail Stop
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATE OF MAILING VIA EXPRESS MAIL

PURSUANT TO 37 C.F.R. § 1.10, I HEREBY CERTIFY THAT I HAVE INFORMATION AND A REASONABLE BASIS FOR BELIEF THAT THIS CORRESPONDENCE WILL BE DEPOSITED WITH THE U.S. POSTAL SERVICE AS EXPRESS MAIL POST OFFICE TO ADDRESSEE, ON THE DATE BELOW, AND IS ADDRESSED TO:

MAIL STOP
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

EXPRESS MAIL LABEL: EV339228432US
DATE OF MAILING: FEBRUARY 26, 2004

SUBMISSION OF PRIORITY DOCUMENT

Dear Sir:

We enclose herewith a certified copy of German patent application 102 54 653.3 filed November 22, 2002 which is the priority document for the above-referenced patent application.

Respectfully submitted,
BAKER BOTTS L.L.P. (023640)

Date: February 26, 2004

By: _____
Andreas H. Grubert
(Limited recognition 37 C.F.R. §10.9)
One Shell Plaza
910 Louisiana Street
Houston, Texas 77002-4995
Telephone: 713.229.1964
Facsimile: 713.229.7764
AGENT FOR APPLICANTS

BUNDESREPUBLIK DEUTSCHLAND

Prioritätsbescheinigung über die Einreichung einer Patentanmeldung

Aktenzeichen:	102 54 653.3
Anmeldetag:	22. November 2002
Anmelder/Inhaber:	Infineon Technologies AG, München/DE
Bezeichnung:	Vorrichtung zur Steuerung der Verarbeitung von Datenwörtern eines Datenstroms
IPC:	G 06 F 9/44

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

München, den 4. Dezember 2003
Deutsches Patent- und Markenamt
Der Präsident
Im Auftrag

Letang

A 9161
06/00
EDV-L

Beschreibung

Vorrichtung zur Steuerung der Verarbeitung von Datenwörtern eines Datenstroms

Die Erfindung betrifft eine Vorrichtung, welche zur Steuerung der Verarbeitung von Datenwörtern eines Datenstroms ausgelegt ist.

In Datennetzen werden häufig Datenströme übertragen, die Datenwörter enthalten, welche von unterschiedlichen Datenquellen stammen. Die Datenwörter der verschiedenen Datenquellen sind dabei in einem Datenstrom sequentiell angeordnet. Die Datenwörter werden in der Reihenfolge, in der sie in dem Datenstrom vorliegen, einer Verarbeitung zugeführt.

Bei der Verarbeitung der Datenwörter kann es sich um die Ausführung sehr einfacher, leichtgewichtiger Befehle handeln. Eine vorgegebene Abfolge derartiger Befehle wird als Thread bezeichnet. Bei der Verarbeitung der Datenwörter eines Datenstroms werden in der Regel verschiedene Threads abgearbeitet. Welcher Thread zu einem Zeitpunkt abgearbeitet wird, hängt in der Regel von Eigenschaften des Datenworts ab, das zu diesem Zeitpunkt an der vordersten Stelle des Datenstroms steht. Insbesondere hängt die Wahl des Threads von der Datenquelle ab, von welcher das zu verarbeitende Datenwort stammt. Dazu sind die Datenwörter häufig mit einer Identifikationsnummer versehen, welche ihre jeweilige Datenquelle kennzeichnet. Anhand dieser Identifikationsnummer kann u.a. der zugeordnete Thread identifiziert werden.

Jedem Thread ist ein eigener Kontext zugewiesen. Der Kontext enthält Informationen über den Zustand, in dem sich die Abarbeitung des Threads zu dem gegenwärtigen Zeitpunkt befindet. Vor Beginn der Ausführung eines Threads kann in den dem Kontext zugeordneten Registern beispielsweise der erste in dem Thread auszuführende Befehl abgelegt sein. Nach Beginn der

Ausführung des Threads können diese Register mit den jeweils aktuellen Befehlen des Threads überschrieben werden.

Zur Verarbeitung der Datenwörter wird eine Vorrichtung benötigt, die dem eingehenden Datenwort den jeweils für es vorgesehenen Befehl aus dem entsprechenden Thread zuordnet und diesen Befehl decodiert, so dass ein dieser Vorrichtung nachgeschaltetes Bauelement, welches die eigentliche Verarbeitung des Datenworts vornimmt, Steuersignale für die Verarbeitung zugeführt werden können.

Herkömmliche Vorrichtungen, die den vorstehend genannten Zweck erfüllen, beruhen entweder auf Software-Lösungen, die auf einem Digitalsignalprozessor ablaufen, oder auf Hardware-Lösungen. Software-Lösungen sind jedoch für Byte-weises Prozessieren eher ineffizient. Hardware-Lösungen weisen den Nachteil einer geringen Flexibilität hinsichtlich Protokolländerungen auf.

Aufgabe der Erfindung ist es, eine Vorrichtung zur Steuerung der Verarbeitung von Datenwörtern eines Datenstroms zu schaffen, welche eine hohe Effizienz und eine hohe Flexibilität aufweist.

Die der Erfindung zugrunde liegende Aufgabenstellung wird durch die Merkmale des Anspruchs 1 gelöst. Vorteilhafte Weiterbildungen und Ausgestaltungen der Erfindung sind in den Unteransprüchen angegeben.

Die erfindungsgemäße Vorrichtung dient zur Steuerung der Verarbeitung von Datenwörtern und umfasst eine erste Einheit, eine zweite Einheit und eine dritte Einheit. In die erfindungsgemäße Vorrichtung geht zu einem Zeitpunkt nicht mehr als ein Datenwort ein. Jedem Datenwort ist ein Thread zugeordnet.

In der ersten Einheit ist der Kontext zu jedem Thread abgelegt. Während eines ersten Taktzyklus wird von der ersten Einheit ein Befehl, der in dem Kontext des dem eingehenden Datenwort zugeordneten Threads abgelegt ist, bereitgestellt.

Die zweite Einheit stellt während eines zweiten Taktzyklus einen Befehl bereit, welcher in der Reihenfolge der Befehle eines vorgegebenen Threads einem vorgegebenen Befehl nachfolgt.

Die dritte Einheit decodiert während des zweiten Taktzyklus den Befehl, der zur Verarbeitung des Datenworts vorgesehen ist, und generiert ein Steuersignal für die Verarbeitung des Datenworts.

Der Befehl, der zur Verarbeitung des Datenworts vorgesehen ist, kann der dritten Einheit in einem vorhergehenden Taktzyklus von der ersten oder der zweiten Einheit zugeführt worden sein. Ferner kann dieser Befehl anhand einer vorgegebenen Reihenfolge von Befehlen oder aufgrund eines Bedingungs-Algorithmus ermittelt worden sein.

Das von der dritten Einheit generierte Steuersignal kann zusammen mit dem Datenwort an ein der erfindungsgemäßen Vorrichtung nachgeschaltetes Bauelement weitergeleitet werden, sodass in dem nachgeschalteten Bauelement die eigentliche Verarbeitung des Datenworts vorgenommen werden kann.

Die erfindungsgemäße Vorrichtung ermöglicht die Steuerung der Datenwortverarbeitung auf eine wesentlich effizientere Weise, als dies mit einem Digitalsignalprozessor möglich ist. Ferner weist die erfindungsgemäße Vorrichtung ein hohes Maß an Flexibilität auf, da viele Parameter, die zum Betrieb der Vorrichtung beitragen, vorgebbar sind.

Die erfindungsgemäße Vorrichtung arbeitet nach dem Prinzip einer Data-Flow-Maschine (data driven). Dies bedeutet, dass

die erfindungsgemäße Vorrichtung nur dann mit der Verarbeitung fortsetzt, wenn eine neues Datenwort in die Vorrichtung eingeht. Im Gegensatz zu einer Data-Flow-Maschine arbeitet eine Von-Neumann-Maschine sequentiell mit jedem Taktzyklus. Dies ist unabhängig davon, ob eine neues Datenwort vorliegt oder nicht.

Die erfindungsgemäße Vorrichtung ist so ausgelegt, dass stets der für die Verarbeitung eines Datenworts benötigte Befehl vorliegt. Dies ist auch dann der Fall, wenn aufgrund eines Threadwechsels ein neuer Kontext aktiviert werden muss. Somit ermöglicht die erfindungsgemäße Vorrichtung die Verarbeitung der eingehenden Datenwörter unter Vermeidung eines Datenstaus (bubble free). Dieses Merkmal weisen die bekannten Multi-Reading-Maschinen nicht auf.

Vorzugsweise ist der von der zweiten Einheit bereitgestellte Befehl der Befehl, der in dem vorgegebenen Thread unmittelbar dem vorgegebenen Befehl nachfolgt.

Zu dem vorstehend genannten Zweck wird die zweite Einheit vorteilhafterweise mit dem Inkrement eines Zählwerts und einem Identifikationswert, welcher einen Thread bezeichnet, gespeist. Anhand des Inkrements und des Identifikationswerts ermittelt die zweite Einheit den Befehl, der in dem durch den Identifikationswert bezeichneten Thread die durch das Inkrement bezeichnete Position einnimmt.

Diese Arbeitsweise der zweiten Einheit hat den Vorteil, dass von ihr stets der Befehl, der in dem betreffenden Thread dem gerade verwendeten Befehl nachfolgt, bereitgestellt wird. Sollte dieser Befehl in dem nachfolgenden Taktzyklus benötigt werden, steht er ohne Verzögerung zur Verfügung.

Gemäß einer vorteilhaften Ausgestaltung der Erfindung wird von der ersten Einheit der Kontext des Threads, der dem in die Vorrichtung eingehenden Datenwort zugeordnet ist, akti-

viert, sofern sich das vorhergehende Datenwort auf einen anderen Thread bezog.

In diesem Fall stellt die erste Einheit vorzugsweise einen in dem aktivierten Kontext angegebenen Befehl des Threads bereit. Dieser Befehl, welcher insbesondere der erste Befehl des Threads sein kann, wird an die dritte Einheit zur Decodierung weitergeleitet. Der zweiten Einheit wird von der ersten Einheit das Inkrement der Position, welche der von ihr bereitgestellte Befehl in dem Thread einnimmt, übermittelt.

Vorteilhafterweise ermittelt die zweite Einheit anhand des erhaltenen Inkrements den Befehl, der in dem Thread dem von der ersten Einheit bereitgestellten Befehl unmittelbar nachfolgt.

Eine besonders bevorzugte Ausgestaltung der Erfindung sieht vor, dass für nacheinander in die Vorrichtung eingehende Datenwörter, denen derselbe Thread zugeordnet ist, solange derselbe Befehl verwendet wird, bis eine vorgegebene Bedingung erfüllt ist.

Während der Wiederholung eines Befehls kann beispielsweise in der dritten Einheit stets dasselbe Steuersignal generiert werden.

Des Weiteren kann vorteilhafterweise die Anzahl der Wiederholungen eines Befehls durch einen Wert vorgegeben sein. Dieser Wert wird bei jeder Wiederholung des Befehls durch die dritte Einheit dekrementiert. Sobald der Wert gleich Null ist, werden die Wiederholungen abgebrochen.

Eine weitere besonders bevorzugte Ausgestaltung der Erfindung ist dadurch gekennzeichnet, dass nach der Erfüllung der vorgegebenen Bedingung für die Verarbeitung des als Nächstes in die Vorrichtung eingehenden Datenworts ein vorgegebener Be-

fehl innerhalb des Threads verwendet wird, falls diesem Datenwort der bereits aktivierte Thread zugeordnet ist.

Dabei erfolgt die Abfrage nach der Erfüllung der vorgegebenen Bedingung vorzugsweise in der dritten Einheit.

Der vorgegebene Befehl, zu dem nach der Erfüllung der vorgegebenen Bedingung gesprungen wird, kann beispielsweise der von der zweiten Einheit bereitgestellte Befehl sein.

Durch eine Verbindungsleitung zur Datenübertragung zwischen der zweiten Einheit und der dritten Einheit ist der von der zweiten Einheit bereitgestellte Befehl vorzugsweise ohne Verzögerungen an die dritte Einheit übermittelbar.

Ferner kann vorgesehen sein, dass der von der zweiten Einheit bereitgestellte Befehl auch an die erste Einheit übermittelt wird und dort in dem Kontext abgelegt wird.

Alternativ zu dem von der zweiten Einheit bereitgestellten Befehl kann nach der Erfüllung der vorgegebenen Bedingung zu einem Befehl, der in der ersten Einheit abgelegt ist, gesprungen werden. Dieser wird dann an die dritte Einheit zur Decodierung weitergeleitet.

Um aus der ersten Einheit einen Befehl auszuwählen, wird vorzugsweise nach der Erfüllung der vorgegebenen Bedingung von der dritten Einheit eine Anweisung an die erste Einheit übermittelt, welche besagt, welcher Befehl bereitgestellt werden soll.

Die vorgegebene Bedingung, deren Erfüllung zum Abbruch der Wiederholungen eines Befehls führt, kann beispielsweise durch ein von außerhalb der Vorrichtung steuerbares Signal oder durch ein bestimmtes in die Vorrichtung eingehendes Datenwort oder durch einen bestimmten Zustand des betreffenden Threads

oder durch einen bestimmten abzuarbeitenden Befehl erfüllt werden.

Eine weitere vorteilhafte Ausgestaltung der Erfindung ist durch einen Programmspeicher gekennzeichnet, in welchem die Befehle zur Verarbeitung von Datenwörtern abgelegt sind. Ferner ist in dem Programmspeicher zu jedem Befehl eine Angabe dazu abgelegt, auf wie viele Datenwörter der betreffende Befehl angewendet werden soll. Der Programmspeicher kann in einer der drei Einheiten der erfindungsgemäßen Vorrichtung angeordnet sein. Insbesondere kann der Programmspeicher Programmzeilen aufweisen, in denen jeweils ein Befehl und die zugehörige Angabe bezüglich der Wiederholungsanzahl abgelegt sind.

Vorteilhafterweise umfasst die erfindungsgemäße Vorrichtung zwei hintereinander geschaltete Verzögerungseinheiten, welche das Datenwort jeweils um einen Taktzyklus verzögern. Da die Vorrichtung zwei Taktzyklen benötigt, um das Steuersignal für die Verarbeitung des Datenworts zu generieren, wird durch die beiden Verzögerungseinheiten gewährleistet, dass das Datenwort und das Steuersignal zeitgleich bei einem der Vorrichtung nachgeschalteten Bauelement eintreffen.

Die Erfindung wird nachfolgend unter Bezugnahme auf die Zeichnung näher erläutert. In der Zeichnung zeigt die einzige Figur ein schematisches Schaltbild eines Ausführungsbeispiels der erfindungsgemäßen Vorrichtung.

In der Figur ist eine Vorrichtung 1 gezeigt, welche eine Kontext-Umschalt-Einheit CS (context switch), eine Befehls-Bereitstellungs-Einheit IF (instruction fetch), eine Befehls-Decodierungs-Einheit ID (instruction decoding), Verzögerungsglieder D1, D2 und D3 sowie Multiplexer MUX1, MUX2 und MUX3 aufweist.

Die vorstehend genannten Bauelemente der Vorrichtung 1 weisen jeweils Eingänge und Ausgänge zur Kommunikation mit den übrigen Bauelementen bzw. zur Steuerung auf. Diese Ein- und Ausgänge werden im Folgenden beschrieben.

Die Kontext-Umschalt-Einheit CS weist Eingänge für einen Kontextidentifikationswert context_id_i, ein Zustandswort id_state_s, einen Kontextidentifikationswert if_context_id_s, ein Befehlssatzdatenwort wb_ir_s, einen spekulativen Zählwert id_spc_s und ein Steuersignal cs_store_s auf. Ausgangsseitig gibt die Kontext-Umschalt-Einheit CS einen spekulativen Zählwert cs_spc_s, ein Befehlssatzdatenwort cs_ir_s und ein Zustandswort cs_state_s aus.

Die Befehls-Bereitstellungs-Einheit IF weist zwei eingangsseitige Verbindungen zu den Ausgängen des Multiplexers MUX1 auf. An ihren Ausgängen gibt die Befehls-Bereitstellungs-Einheit IF ein Befehlssatzdatenwort if_ir_s und einen Kontextidentifikationswert if_context_id_s aus.

Die Befehls-Decodierungs-Einheit ID weist zwei eingangsseitige Verbindungen zu den Ausgängen des Multiplexers MUX2 und jeweils eine eingangsseitige Verbindung zu einem Ausgang des Multiplexers MUX1 und zu dem Ausgang des Verzögerungsglieds D2 auf. Ferner speisen die Kontextidentifikationswerte if_context_id_s und cs_context_id_s sowie ein Nutzdatenwort data_s die Befehls-Decodierungs-Einheit ID. An ihren Ausgängen gibt die Befehls-Decodierungs-Einheit ID den spekulativen Zählwert id_spc_s, das Steuersignal cs_store_s, ein Steuersignal cs_ir_select_s, ein Befehlssatzdatenwort id_ir_s, das Zustandswort id_state_s, das Steuersignal if_source_s und ein Steuersignal dec_o aus.

An die Eingänge des Multiplexers MUX1 sind die Leitungen für den spekulativen Zählwert id_spc_s, den Kontextidentifikationswert if_context_id_s, den spekulativen Zählwert cs_spc_s und den Kontextidentifikationswert cs_context_id_s geschal-

tet. Ein Steuereingang des Multiplexers MUX1 ist mit dem Steuersignal if_source_s beaufschlagt.

An die Eingänge des Multiplexers MUX2 sind Leitungen für das Befehlssatzdatenwort if_ir_s, das Zustandswort id_state_s, das Befehlssatzdatenwort cs_ir_s und das Zustandswort cs_state_s geschaltet. Ein Steuereingang des Multiplexers MUX2 ist mit dem Steuersignal if_source_s beaufschlagt.

Der Multiplexer MUX3 wird eingangsseitig von den Befehlssatzdatenwörtern if_ir_s und id_ir_s gespeist und gibt ausgangsseitig das Befehlssatzdatenwort wb_ir_s aus. Der Multiplexer MUX3 wird von dem Steuersignal cs_ir_select_s gesteuert.

Das Verzögerungsglied D1 wird eingangsseitig von dem Kontextidentifikationswert context_id_i gespeist und gibt ausgangsseitig den Kontextidentifikationswert cs_context_id_s aus.

Das Verzögerungsglied D2 wird von einem Nutzdatenwort data_i gespeist und gibt ausgangsseitig das Nutzdatenwort data_s aus, mit welchem das Verzögerungsglied D3 gespeist wird. Das Verzögerungsglied D3 gibt ein Nutzdatenwort data_o aus.

Ein Taktsignal clk_i speist die Kontext-Umschalt-Einheit CS, die Befehls-Bereitstellungs-Einheit IF, die Befehls-Decodierungs-Einheit ID und die Verzögerungsglieder D1, D2 und D3.

Im Folgenden wird die grundsätzliche Funktionsweise der einzelnen Bauelemente der Vorrichtung 1 beschrieben.

Die Nutzdatenwörter data_i, deren Verarbeitung durch die Vorrichtung 1 gesteuert wird, stammen ursprünglich von verschiedenen Datenquellen und gehen in die Vorrichtung 1 seriell ein. Der Eingang der Nutzdatenwörter data_i in die Vorrichtung 1 ist derart ausgelegt, dass nicht mehr als ein Nutzda-

tenwort data_i in einem Taktzyklus zur Verarbeitung zur Verfügung steht.

Jede Datenquelle ist einem Thread zugeordnet, wobei jeder Thread seinen eigenen Kontext besitzt. Durch den Kontextidentifikationswert context_id_i wird der Kontext-Umschalt-Einheit CS mitgeteilt, zu welchem Kontext das zur Verarbeitung bereitstehende Nutzdatenwort data_i gehört.

Die Kontext-Umschalt-Einheit CS enthält einen Speicher, in welchem die Informationen über den Kontext jedes Threads und der erste Befehl jedes Threads abgelegt sind. Ferner sind in den Speicherregistern jedes Kontexts weitere Befehle abgelegt, auf die durch einen erzwungenen Sprung zugegriffen werden kann. Durch den Kontextidentifikationswert context_id_i wird in der Kontext-Umschalt-Einheit CS der Kontext und der erste Befehl desjenigen Threads aktiviert, auf den sich das Nutzdatenwort data_i bezieht.

Von der Kontext-Umschalt-Einheit CS wird mittels des Befehlssatzdatenworts cs_ir_s der Befehl an die Befehls-Decodierungs-Einheit ID übermittelt, durch welchen der auszuführende Verarbeitungsschritt des Nutzdatenworts data_i bestimmt ist. Ferner erhält die Befehls-Decodierungs-Einheit ID durch das Zustandswort cs_state_s weitere Informationen, die in den betreffenden Registern des Kontexts abgelegt sind und sich auf den aktuellen Zustand des zum Ablauf vorgesehenen Programms beziehen.

Damit das Befehlssatzdatenwort cs_ir_s und das Zustandswort cs_state_s die Befehls-Decodierungs-Einheit ID erreichen, muss sich der Multiplexer MUX2 in dem logischen Zustand 0 befinden. Dazu muss das Steuersignal if_source_s deaktiviert sein.

Ferner geht in die Befehls-Decodierungs-Einheit ID der Kontextidentifikationswert cs_context_id_s ein, welcher dem um

einen Taktzyklus verzögerten Kontextidentifikationswert context_id_i entspricht.

Der durch das Befehlssatzdatenwort cs_ir_s übermittelte Befehl wird von der Befehls-Dekodierungs-Einheit ID decodiert. Als Ergebnis der Decodierung wird das Steuersignal dec_o von der Befehls-Decodierungs-Einheit ID ausgegeben. Das Steuersignal dec_o stellt ein Ausgangssignal der Vorrichtung 1 dar und dient dazu, eine Einheit, die der Vorrichtung 1 nachgeschaltet ist und in der die von der Vorrichtung 1 verarbeiteten Befehle ausgeführt werden sollen, zu steuern.

Des Weiteren werden von der Befehls-Decodierungs-Einheit ID das Steuersignal cs_store_s, der spekulative Zählwert id_spc_s, das Zustandswort id_state_s und das Befehlssatzdatenwort id_ir_s an die Kontext-Umschalt-Einheit CS übermittelt. Für die Übermittlung des Befehlssatzdatenworts id_ir_s muss der Multiplexer MUX3 auf den logischen Zustand 0 geschaltet sein. Dazu wird das Steuersignal cs_ir_select_s von der Befehls-Decodierungs-Einheit ID deaktiviert.

Mittels des Steuersignals cs_store_s wird der Kontext-Umschalt-Einheit CS mitgeteilt, dass Daten von der Befehls-Decodierungs-Einheit ID oder von der Befehls-Bereitstellungs-Einheit IF an die Kontext-Umschalt-Einheit CS weitergegeben werden und dort abgespeichert werden sollen.

Auf den spekulativen Zählwert id_spc_s wird weiter unten näher eingegangen.

Das Zustandswort id_state_s enthält Informationen über den Zustand, in welchem sich die Abarbeitung des Threads am Ausgang der Befehls-Decodierungs-Einheit ID befindet.

Das Befehlssatzdatenwort id_ir_s beinhaltet Informationen, um die Register, die den Befehlssatz des gegenwärtig aktivierten Threads enthalten, zu überschreiben.

Die Befehls-Decodierungs-Einheit ID generiert ferner das Steuersignal if_source_s, mit welchem die Multiplexer MUX1 und MUX2 gesteuert werden.

Die Befehls-Bereitstellungs-Einheit IF erhält den spekulativen Zählwert cs_spc_s und den Kontextidentifikationswert cs_context_id_s, sofern sich der Multiplexer MUX1 in dem logischen Zustand 0 befindet. Dazu muss das Steuersignal if_source_s deaktiviert sein.

Auf den spekulativen Zählwert cs_spc_s wird weiter unten näher eingegangen.

Die Befehls-Bereitstellungs-Einheit IF enthält einen Programmspeicher. Anhand des spekulativen Zählwerts cs_spc_s und des Kontextidentifikationswerts cs_context_id_s wird der Befehlssatz aus dem Programmspeicher ausgewählt, welcher möglicherweise in dem nächsten Taktzyklus benötigt wird. Dieser Befehlssatz kann mittels des Befehlssatzdatenworts if_ir_s in Form eines Maschinencodes an die Kontext-Umschalt-Einheit CS und an die Befehls-Decodierungs-Einheit ID weitergeleitet werden. Zur Weiterleitung des Befehlssatzdatenworts if_ir_s an die Kontext-Umschalt-Einheit CS muss das Steuersignal cs_ir_select_s aktiviert sein, damit sich der Multiplexer MUX3 in dem logischen Zustand 1 befindet.

Ferner wird von der Befehls-Bereitstellungs-Einheit IF der Kontextidentifikationswert if_context_id_s ausgegeben. Der Kontextidentifikationswert if_context_id_s entspricht dem um einen Taktzyklus verzögerten Kontextidentifikationswert cs_context_id_s.

Nachfolgend werden das Zusammenwirken der einzelnen Bauelemente der Vorrichtung 1 und somit die Funktionsweise der Vorrichtung 1 detaillierter beschrieben.

Die Befehlsstruktur, welche den von der Vorrichtung 1 zu bearbeitenden Befehlen zugrunde liegt, lautet:

repeat X until Y else go to Z (1)

Dabei stehen X für einen Befehl, Y für eine Bedingung und Z für eine statische Zielangabe. In dem vorliegenden Ausführungsbeispiel der erfindungsgemäßen Vorrichtung weist ein Befehlssatz die vorstehend genannten drei Angaben auf. Die Befehlssätze werden von den Befehlssatzdatenwörtern cs_ir_s, id_ir_s und if_ir_s übertragen.

Der als Erster in einem Thread abzuarbeitende Befehlssatz ist in den Registern des zugehörigen Kontexts in der Kontext-Umschalt-Einheit CS abgelegt. Der Kontext wird mittels des Kontextidentifikationswerts context_id_i aktiviert, und der in den Registern abgelegte Befehlssatz wird mittels des Befehlssatzdatenworts cs_ir_s an die Befehls-Decodierungs-Einheit ID weitergeleitet. Die Befehls-Decodierungs-Einheit ID decodiert den in dem Befehlssatz enthaltenen Befehl und generiert daraus das Steuersignal dec_o.

Parallel zu dem soeben beschriebenen Vorgehen durchläuft das Nutzdatenwort data_i die Verzögerungsglieder D2 und D3, wobei es in jedem der beiden Verzögerungsglieder D2 und D3 um jeweils einen Taktzyklus verzögert wird. Da das Aktivieren des Kontexts sowie das Decodieren des entsprechenden Befehls jeweils die Zeitdauer eines Taktzyklus benötigt, wird durch die Verzögerung des Datenworts in den Verzögerungsgliedern D2 und D3 sichergestellt, dass das Nutzdatenwort data_i in Form des zeitverzögerten Nutzdatenworts data_o gleichzeitig mit dem Steuersignal dec_o, das aus der Decodierung des entsprechenden Befehls hervorgegangen ist, zu dem nachgeschalteten Bauelement gelangt. In dem nachgeschalteten Bauelement kann anschließend anhand des Steuersignals dec_o die Verarbeitung des Nutzdatenworts data_o ausgeführt werden.

Während des Taktzyklus, in welchem der zu dem aktivierten Kontext gehörige Befehlssatz an die Befehls-Dekodierungs-Einheit ID weitergeleitet wird, wird des Weiteren der spekulative Zählwert cs_spc_s an die Befehls-Bereitstellungs-Einheit IF übertragen. Der spekulative Zählwert cs_spc_s gibt das Inkrement der Stellung des an die Befehls-Decodierungs-Einheit ID weitergeleiteten Befehls innerhalb des zugehörigen Threads an. Wird beispielsweise an die Befehls-Decodierungs-Einheit ID der erste Befehl des Threads weitergeleitet, so ist der spekulative Zählwert cs_spc_s gleich 2. Anhand des spekulativen Zählwerts cs_spc_s und des Kontextidentifikationswerts cs_context_id_s ermittelt die Befehls-Bereitstellungs-Einheit IF in ihrem Programmspeicher den zu dem spekulativen Zählwert cs_spc_s korrespondierenden Befehlssatz. In dem vorstehend genannten Beispiel wäre dies der Befehlssatz, der den zweiten in dem Thread abzuarbeitende Befehl enthält. Mittels des Befehlssatzdatenworts if_ir_s kann der ermittelte Befehlssatz an die Kontext-Umschalt-Einheit CS weitergeleitet werden.

Damit das Befehlssatzdatenwort if_ir_s in Form des Befehlssatzdatenworts wb_ir_s von der Befehls-Bereitstellungs-Einheit ID zu der Kontext-Umschalt-Einheit CS gelangt, muss der Multiplexer MUX3 durch eine Aktivierung des Steuersignals cs_ir_select_s in den logischen Zustand 1 gebracht werden.

Da die Befehlsstruktur der Vorrichtung 1 auf einer Wiederholungsschleife gemäß obiger Befehlsstruktur (1) basiert, kann es sein, dass der mittels des Befehlssatzdatenworts if_ir_s zur Verfügung gestellte Befehl nicht benötigt wird, sondern dass der vorherige Befehl noch einmal ausgeführt wird. In diesem Fall wird der von der Befehls-Bereitstellungs-Einheit IF ermittelte Befehlssatz verworfen.

Bei jeder Wiederholung eines Befehls wird von der Befehls-Decodierungs-Einheit ID das Steuersignal dec_o erneut ausgegeben. Die Wiederholung eines Befehls wird solange durchge-

führt, bis die Bedingung Y erfüllt ist. Die Bedingung Y kann an unterschiedlichste Ereignisse gekoppelt sein.

Bei der erneuten Durchführung eines Befehls werden das Steuersignal cs_ir_select_s deaktiviert und das Steuersignal cs_store_s aktiviert. Durch die Deaktivierung des Steuersignals cs_ir_select_s wird das Befehlssatzdatenwort id_ir_s von dem Multiplexer MUX3 durchgestellt und gelangt in Form des Befehlssatzdatenworts wb_ir_s zur Kontext-Umschalt-Einheit CS. Das Befehlssatzdatenwort id_ir_s enthält den bereits ausgeführten und noch einmal zu wiederholenden Befehl. Das Steuersignal cs_store_s zeigt der Kontext-Umschalt-Einheit CS an, dass sie das Befehlssatzdatenwort id_ir_s abspeichern soll.

Zur Steuerung einer Wiederholungsschleife bieten sich verschiedene Möglichkeiten an. Zum Beispiel kann die Abbruchbedingung Y einer Wiederholungsschleife durch die Erfüllung der Gleichung "count = 0" gegeben sein. Der Wert count ist dabei in dem Befehlssatz enthalten und gibt zu Beginn der Wiederholungsschleife die Anzahl der auszuführenden Wiederholungen an. Bei jedem Durchgang des Befehlssatzes durch die Befehls-Decodierungs-Einheit ID wird der Wert count dekrementiert. Der dadurch aktualisierte Befehlsatz wird mittels des Befehlssatzdatenworts id_ir_s an die Kontext-Umschalt-Einheit CS übermittelt und dort anstelle des bisherigen Befehlssatzes abgespeichert. Sobald der Wert count den Wert 0 erreicht hat, ist die Bedingung Y erfüllt und die Wiederholungsschleife wird abgebrochen.

Alternativ zu dem vorstehenden Beispiel kann die Bedingung Y auch durch eine externe Vorgabe oder durch den Eingang eines bestimmten Nutzdatenworts data_i oder durch das Vorliegen eines bestimmten Zustands des Programms, welcher sich in den Angaben des Zustandsworts cs_state_s oder id_state_s widerspiegelt, erfüllt werden.

Sobald die Bedingung Y erfüllt ist, springt die Abarbeitung des Threads zu der Zielangabe Z. Es handelt sich folglich um einen bedingten Sprungbefehl. Die Zielangabe Z bezieht sich stets auf einen Befehl aus dem aktuellen Thread.

Die Bedingung Y wird in der Befehls-Decodierungs-Einheit ID überprüft. Von dort aus werden auch sämtliche Steuersignale eingestellt.

Sofern das Ziel Z den in dem Thread nachfolgenden Befehl, welcher in dem Befehlssatzdatenwort if_ir_s enthalten ist, betrifft, wird das Steuersignal if_source_s aktiviert, sodass das Befehlssatzdatenwort if_ir_s von der Befehls-Bereitstellungs-Einheit IF über den Multiplexer MUX2 direkt an die Befehls-Decodierungs-Einheit ID übermittelt wird und der darin enthaltene Befehl decodiert werden kann. Dadurch dass sich der Multiplexer MUX1 in dem logischen Zustand 1 befindet, wird an die Befehls-Bereitstellungs-Einheit IF der spekulative Zählwert id_spc_s übertragen. Der spekulative Zählwert id_spc_s veranlasst die Befehls-Bereitstellungs-Einheit IF, in dem anschließenden Taktzyklus den nachfolgenden Befehl zu dem soeben an die Befehls-Decodierungs-Einheit ID übermittelten Befehl bereitzustellen.

Ferner werden in diesem Fall die Steuersignale cs_ir_select_s und cs_store_s aktiviert, sodass der in den Registern des betreffenden Kontexts abgespeicherte Befehlssatz mit dem von dem Befehlssatzdatenwort if_ir_s bereitgestellten Befehlssatz überschrieben wird.

Sofern das Ziel Z eine Zieladresse angibt, deren zugehöriger Befehl sich in der Kontext-Umschalt-Einheit CS befindet, wird dieser Befehl mittels des Zustandsworts id_state_s von der Kontext-Umschalt-Einheit CS abgeholt und mittels des Befehlssatzdatenworts cs_ir_s an die Befehls-Decodierungs-Einheit ID übermittelt. Dazu muss das Steuersignal if_source_s deaktiviert sein.

Zusammenfassend ist nachfolgend ein Algorithmus aufgeführt, welcher in der Befehls-Decodierungs-Einheit ID abläuft und aus welchem hervorgeht, anhand welcher Kriterien die Steuersignale cs_ir_select_s, cs_store_s und if_source_s eingestellt werden:

```
If (Nachfolgender Befehl des Threads wird benötigt) then
     set cs_store_s to active
     set cs_ir_select_s to active
     if (cs_context_id_s ≠ if_context_id_s) then
          set if_source_s to inactive
     else
          set if_source_s to active
     end if
else
     set if_source_s to inactive
     if (Befehlssatzregister werden überschrieben) then
          set cs_store_s to active
          set cs_ir_select_s to inactive
     else
          set cs_ir_select_s to don't care
          set cs_store_s to inactive
     end if
end if
```

Bei dem vorstehenden Algorithmus ist zu beachten, dass die Bedingung „Befehlssatzregister werden überschrieben" auch dann wahr ist, wenn ein in der Kontext-Umschalt-Einheit CS abgelegter Befehl benötigt wird.

Patentansprüche

1. Vorrichtung (1) zur Steuerung der Verarbeitung von Datenwörtern (data_i), wobei jedem Datenwort (data_i) ein Thread zugeordnet ist und zu einem Zeitpunkt nicht mehr als ein Datenwort (data_i) in die Vorrichtung (1) eingeht, mit

- einer ersten Einheit (CS), in welcher der Kontext zu jedem Thread abgelegt ist und welche während eines ersten Taktzyklus einen Befehl (cs_ir_s), der in dem Kontext des dem eingehenden Datenwort (data_i) zugeordneten Threads abgelegt ist, bereitstellt,
- einer zweiten Einheit (IF), welche während eines zweiten Taktzyklus einen Befehl (if_ir_s), welcher in der Reihenfolge der Befehle eines vorgegebenen Threads einem vorgegebenen Befehl nachfolgt, bereitstellt, und
- einer dritten Einheit (ID), welche während des zweiten Taktzyklus den Befehl, der zur Verarbeitung des Datenworts (data_i) vorgesehen ist, decodiert und ein Steuersignal (dec_o) für die Verarbeitung des Datenworts (data_i) bereitstellt.

2. Vorrichtung (1) nach Anspruch 1,
d a d u r c h g e k e n n z e i c h n e t,

- dass der von der zweiten Einheit (IF) bereitgestellte Befehl (if_ir_s) der Befehl ist, dessen Position in der Reihenfolge der Befehle des vorgegebenen Threads das Inkrement der Position des vorgegebenen Befehls ist.

3. Vorrichtung (1) nach Anspruch 2,
d a d u r c h g e k e n n z e i c h n e t,

- dass die zweite Einheit (IF) mit dem Inkrement (cs_spc_s; id_spc_s) eines Zählwerts und einem Identifikationswert (cs_context_id_s; if_context_id_s), welcher einen Thread bezeichnet, gespeist wird, und
- dass die zweite Einheit (IF) anhand des Inkrements (cs_spc_s; id_spc_s) und des Identifikationswerts (cs_context_id_s; if_context_id_s) den Befehl (if_ir_s)

ermittelt, der in dem durch den Identifikationswert (cs_context_id_s; if_context_id_s) bezeichneten Thread die durch das Inkrement (cs_spc_s; id_spc_s) bezeichnete Position einnimmt.

4. Vorrichtung (1) nach einem oder mehreren der vorhergehenden Ansprüche,
d a d u r c h g e k e n n z e i c h n e t,
- dass die erste Einheit (CS) den Kontext des dem eingehenden Datenwort (data_i) zugeordneten Threads aktiviert, sofern sich das vorhergehende Datenwort (data_i) auf einen anderen Thread bezog.

5. Vorrichtung (1) nach Anspruch 4,
d a d u r c h g e k e n n z e i c h n e t,
- dass die erste Einheit (CS) einen in dem aktivierten Kontext angegebenen Befehl (cs_ir_s) des Threads bereitstellt und diesen Befehl (cs_ir_s), welcher insbesondere der erste Befehl des Threads ist, an die dritte Einheit (ID) zur Decodierung übermittelt,
- dass die erste Einheit (CS) das Inkrement (cs_spc_s) der Position, welche der von ihr bereitgestellte Befehl (cs_ir_s) in dem Thread einnimmt, an die zweite Einheit (IF) übermittelt.

6. Vorrichtung (1) nach Ansprüchen 3 und 5,
d a d u r c h g e k e n n z e i c h n e t,
- dass die zweite Einheit (IF) den Befehl (if_ir_s) ermittelt, der in dem Thread dem von der ersten Einheit (CS) bereitgestellten Befehl (cs_ir_s) nachfolgt.

7. Vorrichtung (1) nach einem oder mehreren der vorhergehenden Ansprüche,
d a d u r c h g e k e n n z e i c h n e t,
- dass für nacheinander in die Vorrichtung (1) eingehende Datenwörter (data_i), denen derselbe Thread zugeordnet

ist, solange derselbe Befehl verwendet wird, bis eine vorgegebene Bedingung erfüllt ist.

8. Vorrichtung (1) nach Anspruch 7,
d a d u r c h g e k e n n z e i c h n e t,
- dass die Wiederholung eines Befehls durch die Bereitstellung desselben Steuersignals (dec_o) durch die dritte Einheit (ID) realisiert wird.

9. Vorrichtung (1) nach Anspruch 7 oder 8,
d a d u r c h g e k e n n z e i c h n e t,
- dass die Anzahl der Wiederholungen eines Befehls durch einen Wert vorgegeben ist,
- dass dieser Wert bei jeder Wiederholung des Befehls durch die dritte Einheit (ID) dekrementiert wird, und
- dass die Wiederholungen bei dem Wert 0 abgebrochen werden.

10. Vorrichtung (1) nach Anspruch 7,
d a d u r c h g e k e n n z e i c h n e t,
- dass nach der Erfüllung der vorgegebenen Bedingung für die Verarbeitung des als Nächstes in die Vorrichtung (1) eingehenden Datenworts (data_i) ein vorgegebener Befehl innerhalb des Threads verwendet wird, sofern diesem Datenwort (data_i) derselbe Thread zugeordnet ist.

11. Vorrichtung (1) nach Anspruch 10,
d a d u r c h g e k e n n z e i c h n e t,
- dass die Abfrage nach der Erfüllung der vorgegebenen Bedingung in der dritten Einheit (ID) erfolgt.

12. Vorrichtung (1) nach Anspruch 10 oder 11,
d a d u r c h g e k e n n z e i c h n e t,
- dass der vorgegebene Befehl der von der zweiten Einheit (IF) bereitgestellte Befehl (if_ir_s) ist.

13. Vorrichtung (1) nach Anspruch 12,
g e k e n n z e i c h n e t d u r c h

- eine Verbindungsleitung zur Datenübertragung zwischen der zweiten Einheit (IF) und der dritten Einheit (ID), durch welche der von der zweiten Einheit (IF) bereitgestellte Befehl (if_ir_s) an die dritte Einheit (ID) übermittelt wird.

14. Vorrichtung (1) nach Anspruch 12 oder 13,
d a d u r c h g e k e n n z e i c h n e t,
- dass der von der zweiten Einheit (IF) bereitgestellte Befehl (if_ir_s) an die erste Einheit (CS) übermittelt wird und dort in dem Kontext abgelegt wird.

15. Vorrichtung (1) nach Anspruch 10,
d a d u r c h g e k e n n z e i c h n e t,
- dass der vorgegebene Befehl (cs_ir_s) von der ersten Einheit (CS) bereitgestellt wird und an die dritte Einheit (ID) zur Decodierung übermittelt wird.

16. Vorrichtung (1) nach Ansprüchen 11 und 15,
d a d u r c h g e k e n n z e i c h n e t,
- dass die dritte Einheit (ID) nach der Erfüllung der vorgegebenen Bedingung eine Anweisung (id_state_s) an die erste Einheit (CS) übermittelt, welcher Befehl (cs_ir_s) bereitgestellt werden soll.

17. Vorrichtung nach einem oder mehreren der Ansprüche 10 bis 16,
d a d u r c h g e k e n n z e i c h n e t,
- dass die vorgegebene Bedingung, deren Erfüllung zum Abbruch der Wiederholungen eines Befehls führt, durch ein von außerhalb der Vorrichtung (1) steuerbares Signal oder durch ein bestimmtes in die Vorrichtung eingehendes Datenwort (data_i) oder durch einen bestimmten Zustand (cs_state_s; id_state_s) des betreffenden Threads oder durch einen bestimmten abzuarbeitenden Befehl (cs_ir_s) erfüllt wird.

18. Vorrichtung (1) nach einem oder mehreren der Ansprüche 7 bis 17,
gekennzeichnet durch
- einen Programmspeicher, in welchem die Befehle zur Verarbeitung von Datenwörtern (data_i) abgelegt sind und in welchem zu jedem Befehl eine Angabe abgelegt ist, auf wie viele Datenwörter der Befehl angewendet werden soll, wobei der Programmspeicher insbesondere Programmzeilen aufweist, in denen jeweils ein Befehl und die zugehörige Angabe bezüglich der Wiederholungsanzahl abgelegt sind.

19. Vorrichtung (1) nach einem oder mehreren der vorhergehenden Ansprüche,
gekennzeichnet durch
- zwei hintereinander geschaltete Verzögerungseinheiten (D2, D3), welche das Datenwort (data_i) jeweils um einen Taktzyklus verzögern.

Zusammenfassung

Vorrichtung zur Steuerung der Verarbeitung von Datenwörtern eines Datenstroms

Die Erfindung betrifft eine Vorrichtung (1) zur Steuerung der Verarbeitung von Datenwörtern (data_i), wobei jedem Datenwort (data_i) ein Thread zugeordnet ist, mit einer ersten Einheit (CS), die während eines ersten Takts einen Befehl (cs_ir_s), der in dem Kontext des einem eingehenden Datenwort (data_i) zugeordneten Threads abgelegt ist, bereitstellt, einer zweiten Einheit (IF), die während eines zweiten Takts einen Befehl (if_ir_s), der in einem vorgegebenen Thread einem vorgegebenen Befehl nachfolgt, bereitstellt, und einer dritten Einheit (ID), die während des zweiten Takts den zur Verarbeitung des Datenworts (data_i) vorgesehenen Befehl decodiert und ein Datenwortverarbeitungssignal (dec_o) generiert.

(Fig. für die Zusammenfassung)

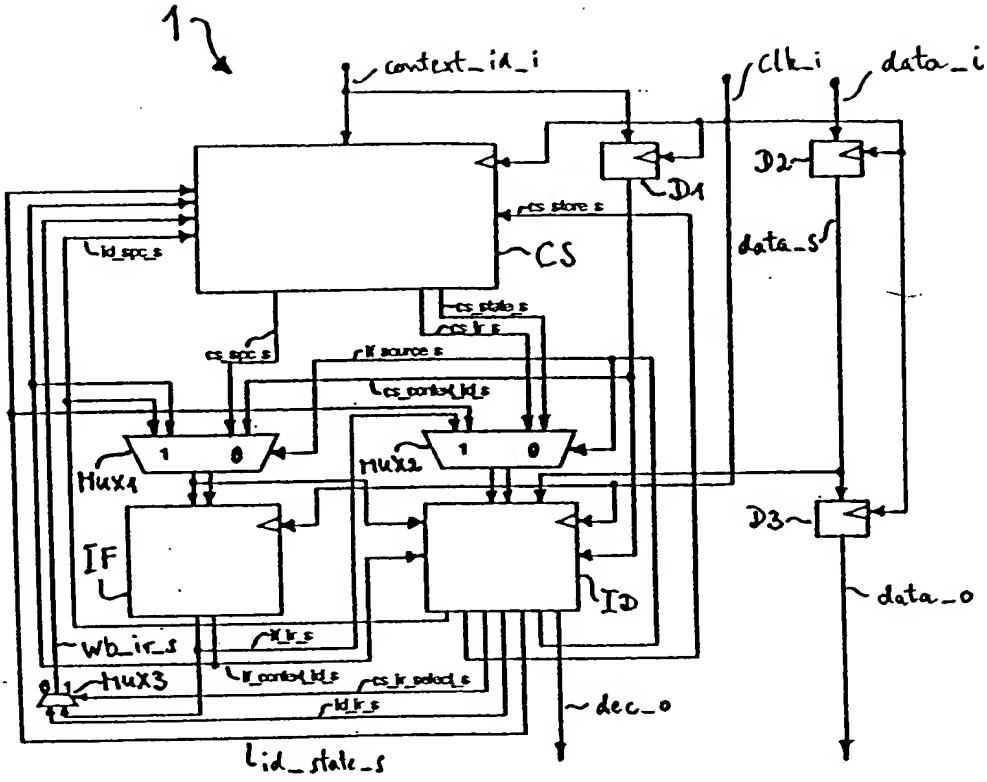
1
context_id_i
Clk_i
data_i
D1
D2
D3
CS
cs_store_s
id_spc_s
cs_spc_s
cs_state_s
cs_ir_s
ir_source_s
cs_context_id_s
data_s
Mux1
Mux2
1
0
IF
ID
data_o
wb_ir_s
ir_s
Mux3
ir_context_id_s
cs_ir_select_s
id_ir_s
dec_o
id_state_s

Fig.

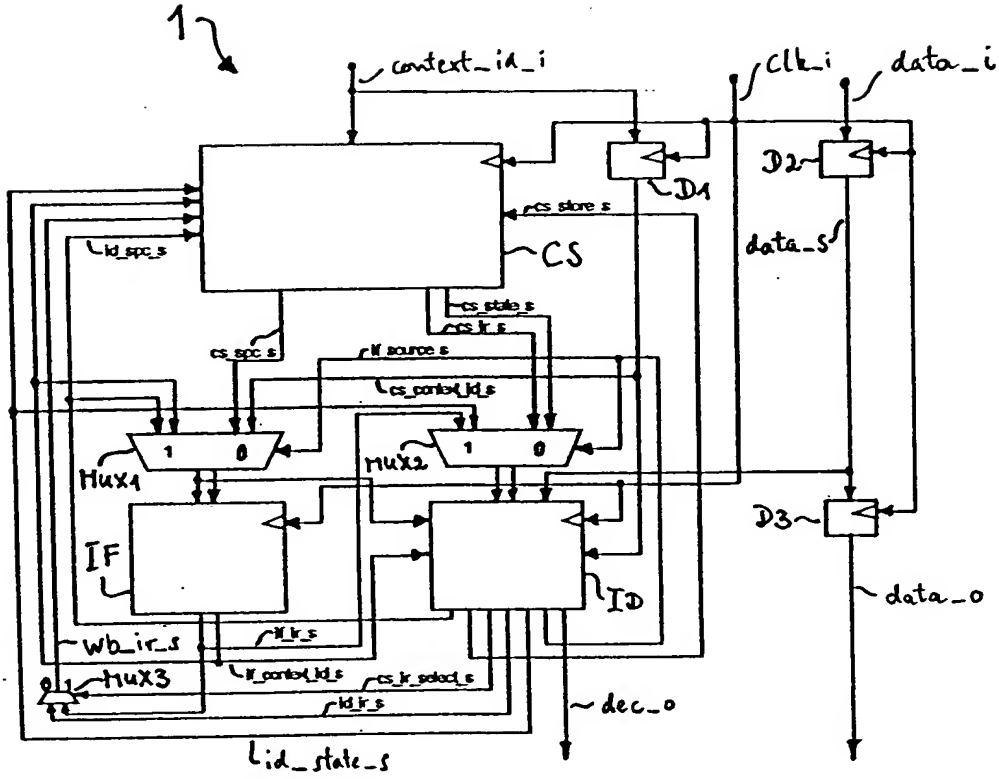
1
context_id_i
Clk_i
data_i
D1
D2
D3
CS
cs_store_s
id_spc_s
cs_spc_s
cs_state_s
cs_ir_s
if_source_s
cs_context_id_s
Mux1
Mux2
IF
ID
data_s
data_o
wb_ir_s
if_ir_s
Mux3
if_context_id_s
cs_ir_select_s
id_ir_s
dec_o
id_state_s

Fig.